OSIDIAL 7

Telephony server
with visual call flow editor

Quick start guide

Installation and user manual

INTRODUCTION

After termination of a setup program the user has access to following application shortcuts placed under OsiDial start menu folder:

Arbo workshop and accompanying tools

ArboWs, all OsiDial resources editor and generator. It is a visual editor for call flow scenarios handling incoming and outgoing calls. It is also used for offline testing of prepared call flows, without a connection to a real telephone line. For this purpose a Windows compatible sound card is required. Calling phone keyboard is simulated on the screen.

EditBase, database editor. This is a tool for editing of OsiDial integrated databases, for storing passwords and/or fax on demand documents.

EditMbx, subscriber editor. It is used for configuration of voice mail boxes of voice mail subscribers.

ViewStat, real time call statistics viewer. It presents graphically the amount of calls versus time of day or day in month. The lasting of all channels is presented as well.

Voice and fax server

OsiDial.exe, this is a run time engine of a whole package. It acts as a windows service. It requires the Donjin boards to be installed and properly configured.

Osc.exe, this is the user interface for osidial service, which alone runs with no user interface at all. Both modules communicate each other, though OsiDial.exe runs in Windows service account and OSC executes under user account. Osc may be started on local console or in the remote desktop session.

OsiDial package other files (selected)

OsiLL.dll, library of common functions used in other package components. It also contains low level functions serving directly installed telephony boards.

OsiStdFunc.dll, library containing all standard functions used for serving calls. These functions are encapsulated as **nodes** on the arbo call flow (see below).

OsiFuncEx.dll, contains extension with custom functions. So called custom nodes. There might be several extension dlls with different filenames, but their structure is always the same. Only one extension dll may be used at a time. This library makes possible using of non standard functions in a call flow scenario without a need of recompiling of the rest of package code.

Osidial.exe, OsiDial service executable. It can also be started as a user mode application. It manages all other modules together. Initializes hardware layer, loads prepared call flaw scenarios, handles telephony events and so on.

Osc.exe, above described viewer module with user interface for osidial service.

INSTALLATION

Arbo resource workshop with acompanying tools and Osidial run time programs can be installed on the same computer or on two different computers.

Hardware configuration for Arbo resource workshop

Arbo resource workshop, for creating and testing of telephony call flow services, requires a multimedia PC. The sound card, compatibel with Windows mutimedia sound interface, coupled with microphone and speaker(s), is enough for preparing audio prompts and offline testing.

Hardware configuration for Osidial run time application

The real, multiline telephony board of one of supported vendors has to be installed in the Windows operating system before starting Osidial setup program. The Windows system version should be apropriate to telephony board being used. OsiDial from its own side is a 32 bits application that runs well on Windows 2000 proffesional or any newer version of NT system branch.

Installing from setup image

Most often the installation is performed by authorized reseller, on site or remote if possible. Installation is initiated by running osidialsetup.exe. Updated version can be downloaded anytime from http://www.ositron.pl/download/osidialsetup.exe. It is recommended to leave the default installation folder : C:\ARBO70. The setup destination folder might be freely changed, however all installed sample call flow scenarios do asume the default OsiDial root folder to be C:\ARBO70. If installed in other directory, one has manually to change directoriy references inside the call flow project file. Call flow project file describes u.a. the location of audio prompts being used.

Installed directories structure

Excluding some system library DLLs, all other files are installed in a target install directory given at the setup time. This way, after installing in a default directory C:\ARBO70, there is created directory and files structure as described below:

C:\ARBO70 contains files of following types:

- .EXE and .DLL components for OSIDIAL application
- .ARB arbo call flaw project files
- .STA statistics files
- .CFG files holding general configuration of vocal server
- .SRV files with call flaw services assigned to IVR channels
- .PBX files with settings related to telephony switches
- .MBX files with data of voice mail subscribers

For a given arbo project (arbo = call flaw scenario) there are present several subdiirectories containing data and audio files, for example if one has a project MESSVOC.ARB:

C:\ARBO70\MESSVOC.ARB	arbo project file
C:\ARBO70\MESSVOC	directory for call flaw functions (node) data and configuration dedicated only to this project (private config data). Often there are placed here the statistics files related to this project.
C:\ARBO70\MESSVOC\ARBOMESS	audio files for this project, for simple nodes
C:\ARBO70\MESSVOC\DATAMESS	additional information (often empty)
C:\ARBO70\MESSVOC\FAX	fax document files
C:\ARBO70\MESSVOC\FUNCMESS	audio files for other function nodes
C:\ARBO70\MESSVOC\MESSBOX	voice mail messages
C:\ARBO70\MESSVOC\NUMBMESS	audios for ordinals, dates and times, money
C:\ARBO70\MESSVOC\STANMESS	audio prompts for voice mail

(Several arbo projects may share same directories, especially FUNCMESS, STANMESS and NUMBMESS. Most often used audio prompts, that potentially used in every arbo project are prerecorded and located under C:\ARBO70\ORIGINAL directory.

Multiple languages concept

Osidial audio prompts may be recorded with several spoken languages. Every language has its own subdirectory named LANGXXX, where XXX is the number of the language. Assigning numbers to particular languages is totally up to the user, but originally prerecorded audios are delivered with following convension : LANG001 for french, LANG002 for english, LANG003 – german and LANG004 – polish. Arbo can change spoken language dynamically at runtime. Language is just a variable indicating which subdirectory the files should be played from. Thus the same arbo nodes (functions) may be used for different languages. And the audio files will be taken from different subdirectory. That makes maintanance of the project much easier.

ARBOWS, RESOURCE WORKSHOP EDITOR

Starting ArboWS

The shortcut for **Resource workshop** editor is placed by a setup program under OsiDial folder in menu Start.

Opening existing arbo project

Projects are named as '**arbo**', because they are represented graphically in a form of tree structure (call flaw diagram, *french : arborescence*). ArboWS makes possible editing and testing one project at the time. OsiDial voice server is of course able to serve multiple different projects on different telephony lines, often referenced as IVR channels as well. The projects or services may be assigned to channels statically or they can be changed dynamically when the call already exists.

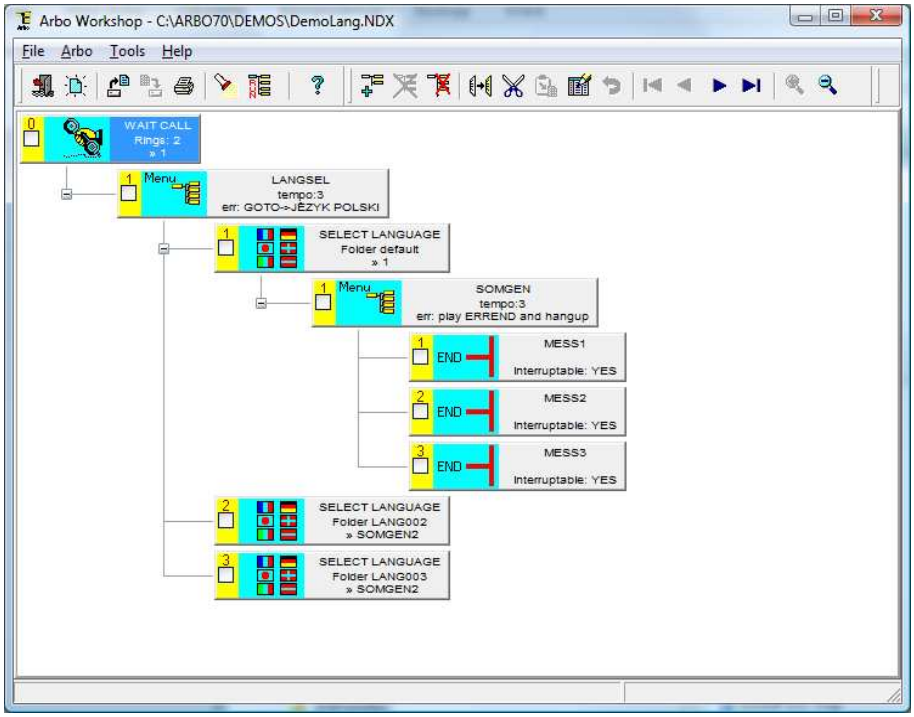
One has to use menu command File/Open, to start working with already existing arboo, for example predefined project **DEMOLANG.ARB**.

A standard Windows file open window will be displayed offering files for read, files with extension **.ARB** . ARB file serves as a 'project file' and contains information about layout of different components of the service. The structure of the service contains arbo tree structure itself as well as several directories with acompanying audio files and other data files.

Po otwarciu serwisu jesteś gotowy do modyfikacji i/lub testowania tego serwisu na swoim komputerze multimedialnym.

Testowanie istniejącego projektu

Należy wybrać **Arbo/Test Lokalny** lub wybierz przycisk z paska narzędziowego.


Arbo Workshop - C:\ARBO70\DEMOS\DemoLang.NDX
File Arbo Tools Help
0 WAIT CALL Rings: 2 » 1
1 Menu LANGSEL tempo:3 err: GOTO-»JĘZYK POLSKI
1 SELECT LANGUAGE Folder default » 1
1 Menu SOMGEN tempo:3 err: play ERREND and hangup
1 END MESS1 Interruptable: YES
2 END MESS2 Interruptable: YES
3 END MESS3 Interruptable: YES
2 SELECT LANGUAGE Folder LANG002 » SOMGEN2
3 SELECT LANGUAGE Folder LANG003 » SOMGEN2

Testing of a service is very simple, the telephone device is simulated by sound card when dealing with voice playing and/or recording, and by numerical keyboard on screen, imitating phone buttons, displayed at the top right corner of the computer desktop. Calling person moves between functions contained in arbo (tree nodes) by responding service audio prompts with phone keys on numerical keyboard. It corresponds exactly to real situation, when a real calling person reacts by pressing phone keys. In both cases arbo service behaves identical and the same set of code is executed. Thats why it is possible to precise test the project before applying it on real life telephony line. Of course there are number of functions that return values independent if the user pressed any key or not. For example a function reading query from database may return 1 on success or 0 in case of a problem. Well designed call flaw has to handle both situations. In case of return value 1 direct the control to child node 1 presenting for example information from databse in audible form, but in case of error, direct control to child node 0 explaining the situation or asking for calling later.

Modifying an existing project

Designing of call flaw scenario is to compose phone call serving project consisting of visual components called nodes, that correspond to available functions. The library of nodes is rich, it counts more than 100 and is being still developed.

From the designing point of view it is important to know, what function is executed by a given particular node and to what child nodes is given control after finishing to execute node. The node may have one or more children, as well in special cases the node may also have no children, like for example a node terminating a call. But in oposite every node has one and only one parent. With only one exception of a root node, which has no parent. In most cases the

root node implements the function of waiting for a new call to be arrived. Even in services for outbound calling, they start with WAITFORCALL node.

To change a node type in an arbo one has to find that node, focus it and then double click on it or use a toolbar button : **Edit data** .

To add a new child node for a given focused node, one has to press a toolbar button : **Add node** .

For simpliest services the user moves throughout the nodes by responding to audio prompts. In such cases call flaw control corresponds to phone key pressed and control goes to child node on position related to phone key. Arbo diagram presents the position of each node with a digit on the left side of node area. The person designing the service determines this postion at a time of adding a new node. Just selects the branch number, where the new node has to be placed.

More advanced nodes may direct controls to child nodes in a way not related to phone keys being pressed, e.q. ‘1’ – if true, ‘0’ - if false, may direct control always to parent node or even may jump to any other node labeled with a given keyword. Most often used rule it ro direct control to child node on position ‘0’ in case of error and on position ‘1’ if the execution was successful.

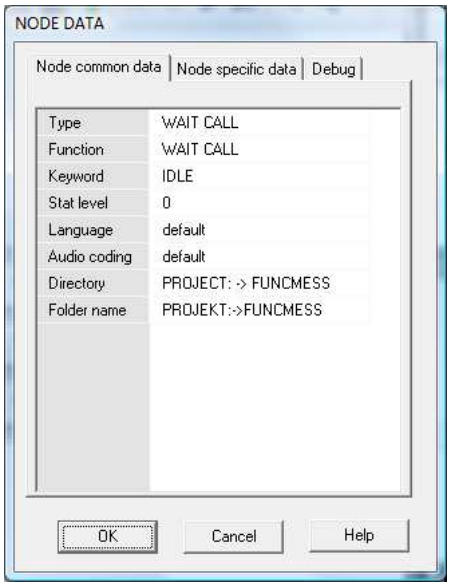
Chapters below describe basic knowledge required for creating and/or modifying of call flow scenarios, used by an OsiDial vocal server.

Node modification

To edit node data (parameters) one has to highlight the node and press a toolbar button or double click on it as described above.

A dialog window 'NODE DATA' will be displayed, with active tab set to: Node common data.

Here are determined the basic data of every node regardless of underlying function.



First of all the required node type has to be selected from a drop down combo box labeled **'Type'**. The list of available node types may be expanded in future versions and also may contain some custom node types, when custom dll modules were ordered.

The field '**Function**' shows short descrition of selected type. Per default it repeats the type name, but may be edited by the user to make an arbo diagram more readible. Exactly this field content is displayed on the tree diagram. For some, very basic node types, the field : **Function** is substituted by a field : **Message**. In this case it contains a .WAV file being used as an audio prompt. Thi file name may be selected from a list or written manually.

After determining node type the tab : '**Node specific data**' has to be switched to, in order to provide parameters suited for a given function. Most parameters have their default values for typical situations.

The meaning of all the fields placed on 'Node common data' tab is described below:

Type determines a node type and the realized function in a service.

Message or

Function field 'Message' for simple nodes or 'Function' for more advanced nodes (beacuse they typically use more than one audio prompt).

Keyword This field may be used to assign a unique label to a node. (to make possible jumping between nodes (GOTO) or for statistical references).

Stat level depending on selected stat level, there are less or more information written at runtime into the statistics files. This field is present mostly for backwards compatybility. It is not taken into account for newly developed node types. All information eventually needed for debug purposes are now savved in audit log files. Best way is to leave the default value of 0, causing for older nodes to write least possiible information, which keeps the server performance high.

'**Language**' and '**Audio coding**' should not be changed in most standard applications. Default audio coding for analog telephone lines is 8000 kHz, 8 bit, μ-Law, mono, and for digital trunks 8000kHz, 8bit, A-Law, mono. Default language for audio prompts depends on selected language during installation. If selected language is english, default language for audio prompts is english as well amd are located under .\LANG002 subdirectory, as described in previous chapters.

Directory determines the location of audio files related to this node. Directories starting with ***PROJECT :->*** correspond to arbo project standard directories described earlier. They may be changed globally for all nodes contained in the arbo. If one selects ***Folder Windows,*** then in field **Folder name** it is possible to choose a fully qualified windows directory, where the IVR will search for audio files. It may even be a shared network directory, for example for recording audios that later need to be accessed remotly.

After switching to 'Node specific data' there are controls displayed that depend on previous selected node type. Under this tab one can determine all other node parameters.

Directions for using ArboWS editor :

- Double click on node area initiates node parameters editor.
- All audio files drop down combo boxes contain the list of audio files present in the given directory. So it is easy to choose an existing audio file. Separate standing fields for audio file name input are most often acompanyied by a **Play/Record** button. Pushing this button initiates and integrated audio player and recorder. Its possible to play the file previously recorded or record it once again. In seldom cases, when the field is in the form of the inplace editor of the string grid, audio playing is possible after pressing right mouse click on the combo box drop down button. **Note** : audio editor allows in addition storing of audio text content. This text makes recording easier from one side, because it is displayed while recording and might be read. But from other side, if somewhen in the future the text to speech engine will be

available for Osidial, it might be used for playing prompts event if they weren't prerecorded before.

- When creating a new project it is good to enter a value of C:\ARBO70\ORIGINAL for a field 'Original folder' (here are located prerecorded audios delivered with OsiDial system) and let the basic files to be copied into a new project directory. It is not recommended to rerecord the audio files placed inside the original directories, because it might have impact for working of greater amount of arbo services, that also might refer to this directory.
- The node may have zero, one or more child nodes, depending on its own node type. Review delivered demo projects for more information.
- It is possible to copy/cut and paste some parts of the tree. Even from one project into the other. The drag'n'drop functionality may be used for this purpose as well. Nodes dragged are handled as moving (cut and paste). Dragging while control key is pressed, works as copying (copy and paste).

Configuration of OsiDial server

The way of working of vocal server is determined by a content of an OsiDial.ini file located in the install root directory, default c:\arbo70. In typical situations only one section requires attention : [OsiDial_for_Windows].

Meaning of variables in section OsiDial_for_Windows is described below:

sLanguage=ENG, defines the language of the user interface for the OsiDial vocal server. In current version only english user interface is implemented. It is not recommended to change this value. In future there might be additional languages implemented as user interface

sMsgLanguage=2. This is the number of default audio prompts language. Default value 2 corresonds to LANG002 subdirectories, where prerecorded files with english language prompts are located.

sPath=c:\arbo70 defines the directory where are stored files with statistical information about handled calls. They are used by a ViewStat application. This directory might be a shared network drive, in which case call statistics may be viewed in real time even on other PC in LAN.

sName=customer name. This variable holds most often the company name where the server is located, identifying the IVR application. This name is also displayed inside the Osidial user interface program OSC.EXE.

sNum=1, holds the number of the server. It is useful in case of several instances of vocal servers, cooperating inside one local area network.

AuditSubDir=audit. It holds the name of the directory where the audit log files are stored. Audit files are text files, with dates and time stamps logging all activities on all channels. They are useful if there are any doubts if the call flaw scenario is executed as expected.

customDll= contains the full filename of the custom DLL containing additional functions, specially designed for not typical requirements. Default value is empty, which means that there is no external custom DLL used.

SRV=osimatic.srv. It holds the name of file with osidial services. This .SRV file has to be placed in a root install directory, default c:\arbo70. It defines arbo services being assigned to IVR channels. It is possible assignment of different scenarios to different channels or changing the service scenario at call time (node **CALL ARBO**). In addition the files .SRV define the ivr channel rights for outbound traffic. These files are prepared by ArboWS resource workshop, menu command Tools/Services.

CFG=osimatic.cfg. Holds the name of the file with general ivr configuration. This file defines the audit log details level, allow for including/excluding particular nodes from logging, determines the maximal call serving time, etc. The file is prepared by ArboWs menu command Tools/Configuration.

MBX=osimatic.mbx. This is the name of the file with voice mail subscribers data. It is used only in special services acting as the voice mail service. This file

is prepared in an external application **EditMbx**, located in c:\arbo70. Predefined demo services, installed with the program, contain the voice mail sample : **DEMOMVOC**. The real scenario may even work better, as there is a possibility to use signaling mode available under many telephone switches, which allow to differentiate between direct calls (to replay messages left in the box before) and redirected calls (after the originally called subscriber does not answer).

INITDELAY=0 holds the time in seconds. After this time OsiDial server application starts initializing of telecommunication hardware. Sometimes, if OsiDial service start mode is set to authomatic, it may happen, that Osidial service is ready to work sooner that the hardware board driver initializes. In such cases the delay has to be increased.

DependentServices= holds the comma separated list of services that has to be started before OsiDial continues to start. If the list is not empty, OsiDial checks if required services are started and if not, they are started. This is useful if IVR relies on any external services that has to be started in order to let the vocal server to work properly.

AutoLaunchViewer=1. This field determines if OsiDial service should start the user interface application OSC.EXE after the user logs into the Windows session. If the value is 0, the viewer is not started authomatically, but the user can always start it manually. Regardless if the viewer is started or not, all audit logging information are stored in the audit directory.

‘SIMPLE’ nodes

NODE simple choice node like for example: "Please press 1, 2 or 3". Field : Message in common node data tab should contain the audio with asking prompt. If file not already exists, just input the name of the new file and then press play/record button on the right side to record the file using PC sound card. Error and timeout messages may be defines after going to Node specific data.

SUMMARY NODE similar to simple 'NODE' with one additional feature: if the ‘*’ key is pressed later when executing one of the child nodes, execution control goes back to upper levels and stopps at the node with feature SUMMARY.

NODE AUTO and NODE SUMMARY AUTO
this node after playing an audio message does not wait for any user input, it just directs control to one of the child nodes or jumps to any node labeled with a given keyword.

NODE GUIDE it is a sample of nodes without any child nodes. After playing an audio message service control just returns to parent node.

END NODE this node may terminate a branch on an arbo tree. But it not terminates the call. After playing an audio message service controls goes back to upper level until the next SUMMARY NODE or SUMMARY NODE AUTO.

END CALL After playing an audio message the call is terminated and service control goes back to node awaiting for a next call.

GOTO jumps the service execution to any node, that was previously labeled with a **keyword**.

PAUSE pauses service execution for a given amount of time. No message is played at this time. Optionally this time may be shorter if a phone key is pressed of fax tone detected.

SELECT LANGUAGE
Changes the spoken language of all subsequently played audio prompts. Corresponding audio files, previously recorded in a given language has to exist on disc.

List of 'FUNCTION' nodes

'BASIC' nodes

WAIT CALL Always set as the root node. Detects an incoming calls and reacts as designed : answers, rejects, not answer etc. This note is also responsible for time sharing. This lets to do some action if there is no call arrived after a defined period of time. This way it is possible to perform some background tasks and/or to prepare an outgoing calls.

WORKING HOURS directs service control to different child nodes depending on time of day and/or day of week. Weekends or well known holidays may also be taken into account.

WELCOME this function is not so frequently used anymore. Its task was asking the calling person to press a '*' key, for verifying if there is an DTMF capable phone on other side. By the way pressing '*' twice often caused switching pulse mode to DTMF mode.

PASSWORD asks for a password and verifies it in the internal database. This database has to be prepared before with program **EditBase32**. The further service execution depends on it, if entered password matched the one given in a table or not.

REMOTE RECORD makes possible recording of audio files over the phone.

SELECT RANDOM directs service controls to a random selected child node.

SPEAK TIME plays current time.

PLAY + << >> For playing longer prompts. Like on any magnet type player it is possible to rewind a piece of audio back or move the playing 'head' forward.

'Phone' nodes

OUT CALL makes an outgoing call.

TRANSFER CALL Directs existing call to another phone switch extension. If it succeeds, the IVR channel becames available for handling next call.

ROUTING NODES bridges two IVR channels over scbus. It is sometimes the only way of transferring calls, if the switch is not

capable of doing it or when the so called ‘music on hold’ is not acceptable for callers.

‘FAX SERVER’ nodes

NODE FAX marks one or more documents as intended for sending by fax.

VOCALFAX BASE this function asks the caller for giving a document reference number. Based on this number, a fax document from fax database is assigned for sending as fax. Fax database is prepared by an external application **EditBase32**.

FAXCALL NUMBER asks the phone number, where the fax has to be sent.

SCHEDULE FAX verifies information gathered by 3 above functions and creates a fax request file, which is used later by executing function for sending outgoing fax.

SENDFAX/TEL sends fax by placing an outgoing call. The target phone number is just called.

SENDFAX/MANUAL uses a currently existing call to send a fax. So the charging for a call is totally on the calling person side.

SENDFAX/LAN uses an external, separate fax server

RECEIVE FAX receives incoming fax

ODŚWIEŻVFX/DANE updates the fax database performing copying of fax documents from a corresponding directory.

ODŚWIEŻVFX/TEL updates the fax database performing fax receiving.

'VOICE MAIL' functions

MV EXTENSION NR asks the subscriber extension number or extracts it from interaction with a telephony switch.

MV WELCOME welcome message at the subscriber’s mail box.

MV PASSWORD checking subscriber’s password.

MV RECORD records a message in the subscriber’s box.

MV LISTMESS anounce the count of messages in the box.

MV PLAY plays messages from the subscriber’s box.

MV USER ADMIN makes the owner's voice mail box configuration.

MV CALLBACK makes an outgoing call at the defined time periods to notify the box owner about new messages left in the box.

MV NOTIFICATION notifies about new messages using phone switch facilities. Typically this makes the LED flashing on box owner's telephone. Alternatively the owner's dial tone is changes.

'SAYING' and 'ARYTHMETIC' nodes

PLAY NUMBER plays the number stored in variable.

PLAY MONEY plays amount of money stored in variable.

PLAY DATE plays date and/or time stored in variable.

SET IVAR assigns value to an integer variable.

ASK IVAR asks the caller for input of integer variable and assigns it to a variable.

SET SVAR assigns a string value (characters sequence) to a string variable.

ASK SVAR asks the caller for input of characters sequence and saves it to a string variable.

OPERATIONS basic calculator operations. Operates on internal variables space of type integer or string.

SPECIAL STATS used to save user designed logs into a text file.

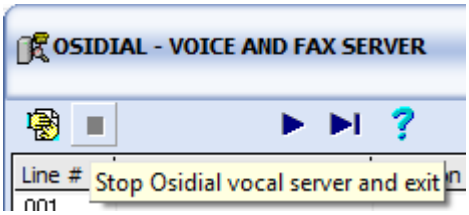
OSIDIAL VOCAL AND FAX SERVER

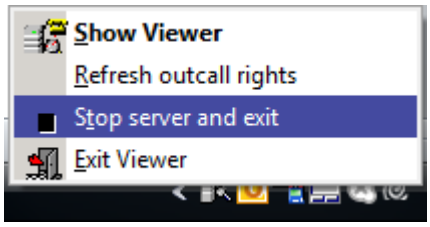
Starting of vocal and fax server

Osidial.exe application can be started both as a typical user mode application or as a regular Windows service. Using as service mode is preferred. The setup program registers Osidial service and configures manual startup mode. It can be changed every time to authomatic startup mode. To start a service, one has to open Windows services management console, find the entry for **OsiDial IVR – vocal and fax server**, and then press the start button. At the time of starting OsiDial IVR application, the telephony hardware driver and any related services should be already started and running well. If there is configured to use an user interface application – OSC.EXE, the related window is displayed, showing information with current execution status on all IVR channels. Every audit log entry saved on disk is also presented on user interface window. After initializing of all hardware channels and all arbo services, the viewer application is minimized and visible only as an icon inside the Windows system tray area. One can always restore the viewer by double clicking this icon or using the right mouse context menu. If Osidial is started as normal user mode application, a dialog box is displayed informing how to initiate the service mode usage. If the user ignores it and press **Continue** button, Osidial will start as typical windows application, not as a service. The difference is among others, that after user logoff the session, the program will stop, while if started as a service, it will keep running.

Stopping the IVR server

If all services work well as expected, there is no need at all to terminate the running server. It is designed for a continuing long term usage. If there is any reason to stop the server, one has simply to stop the service : **OsiDial IVR – vocal and fax server**. In user mode application there should be a toolbar button used or a context menu command: **Stop server and exit**.





INSTALLED ARBO SAMPLES

Osidial IVR comes with a rich set of example services. Their names start with DEMO and their data and audio files are located under subdirectory names DEMOS. However sample services may use audio files located under ORIGINAL subdirectory. Note that examples are not always full services, sometimes all possible errors are not handled. Also provided audio files are for demonstrating purposes only. They may contain different voices, grammar or pronounciation errors. Even some audio prompts may be missing. They ilustrate most of the possibilities of OsiDial. Demo samples may be easily tested offline, under Arbo workshop. Nevertheless the examples are workable and may be used with no problems on real telephone lines.

Overview of installed demos

The examples are listed in our opinion from simpliest to more complex. So, this allow to learn the possibilities of Osidial even without having any experience with it in the past.

DEMO1 This is the simpliest script. After answering a call, the system asks for simple choice of pressing phone key : 1, 2 or 3. After receiving user input, the apropriate message is played with corresponding information. Not allowed keys or timeout errors are handled as well.

DEMO2 This is more complicated example based on DEMO1. After saying welcome to the calling person, the system checks current day of week and current time of day and directs control depending on it. Sometimes it may disconnect and at other times it may present the menu from previous sample. To change the way, the IVR handles days and times of day, one has to edit COK.DHM file using menu command: **Tools/Hours editor**.

DEMO3 Shows how to organize a hidden (not official) entry into a service. If a user during a welcome message enters 9 key, the unofficial part of the service may be presented. Real script might contain here in addition a kind of authentication.

DEMO1DYNAMIC Similar example to DEMO1. But this time the menu prompt is not hard coded. The name of the file is stored in the variable. Real service could read this variable for example from a database.

DEMOACD This arbo shows the possibility of organizing of call queues using IVR channels. Waiting in queue users can hear theit position in queue and estimated time, in minutes, of staying in queue.

DEMOBANK Requires special database access and a custom module, so it is not usable in standard version. One can view similar possibilites but without database acces after opening **DEMOSENTENCE** example.

DEMOCALLARBO Demonstrate how to change arbo at call time. The common practical way of using this feature is to load arbo basing on called number. Thus an IVR can serve multiple scenarios fylly dynamic.

DEMOPWD Demonstrrates access to service information only after providing a valid password. Here valid passwords are 1234 and 0519. Password database was prepared using Internal database editor.

DEMOSENTENCE Shows several nice possibilities of 'talking' functions of Osidial. There are played information about current date and/or time. As well as a current account balance. All information are taken from variables, so in real app might be for example obtained from database.

DEMOOUTCALL Presents the outgoing call service.

DEMOMVOC quite advanced example of voice mail system. In real application some information may be received during interaction with telephony switch. This sample asumes the worst case : no interaction. Whe leaving the message, one has to enter the extension number : 10 or 18 and leave the message. To hear recorded messages one has again to enter the extension number and press a secret key : 7 during playing the mailbox welcome message. After asking a password, one should enter 8642 as mailbox password. Then the content of the mailbox will be presented. All subscriber's data may be prepared or changed using **EditMbx** subscriber editor.

DEMOSVAR Waits for input of a maximum 10 digits string. After receiving a sequence, the IVR plays it. It uses formating 3.3.2.2, it means the digits are grouped similary like phone numbers and played as partial numbers : xxx-xxx-xx-xx. This method is also well

suited for playing for example bank account numbers or credit card numbers and so on.

DEMONUM1 Asks a 3 digits number and then plays it.

DEMONUM2 As previous example, asks a 3 digits number and plays it. After that there is offered a possibility to enter a 2nd one digit number. This number will be added to previous one and the result will be played. Subsequently adding and playing new result will be repeated until the result reaches or exceeds 999.

DEMOCALC Just asks the first 3 digits number and stores it in the variable. Then asks the 2nd 2 digits number and stores it in another variable. After that the sum of both values is calculated and the result is played.